Estimating Invasion Time in Real Landscapes: Supplementary Materials

Daniyah A. Aloqalaa[1], Jenny A. Hodgson[2],
Dariusz R. Kowalski[1], and Prudence W.H. Wong[1]

[1]Department of Computer Science, University of Liverpool, UK
{d.a.aloqalaa, d.kowalski, pwong}@liverpool.ac.uk
[2]Department of Evolution, Ecology and Behaviour, University of Liverpool, UK
jenny.hodgson@liverpool.ac.uk

1 Invasion Network

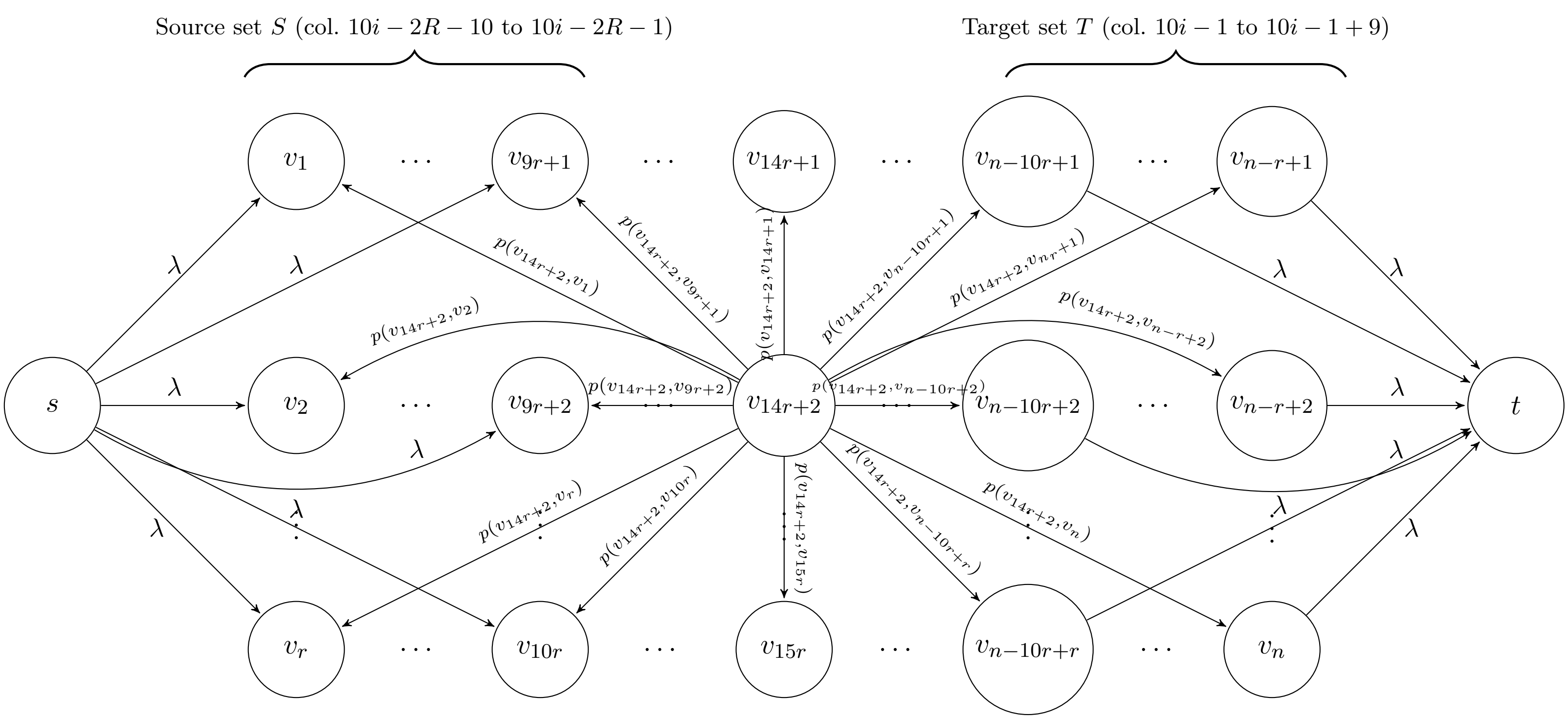

Figure S1: The constructed invasion network N for the sub-landscape graph G' of a given graph G of size $H \times W$. Each vertex (except s and t) is connecting to all other vertices by edges and given weights which is equal to the transition probabilities p between the patches, as shown by vertex v_{14r+2}.

2 Proofs

Below we re-state and prove Theorem 4.2 from the main paper.

Theorem 4.2 *For any landscape graph G, the number of rounds in an asynchronous execution of invasion protocol is asymptotically not bigger than the number of rounds in a synchronous execution of invasion protocol plus $\log n$, with high probability. On the other hand, the expected number of rounds in a synchronous execution of invasion protocol on landscape graph G is $O(\gamma(N) + \log n)$, where N is the invasion landscape network of landscape graph G.*

Proof Consider an asynchronous execution of invasion protocol. Partition asynchronous steps into consecutive rounds. Consider a modified invasion protocol, executed in asynchronous model, in which each vertex remembers whether it has already populated some unpopulated vertex in the current round or not; if it has, it skips any other possible action in this round even if it is selected by the asynchronous mechanism (i.e., by the model feature that randomly schedules edges to execute their populating operation in the execution). This protocol is clearly not faster than the original invasion protocol in asynchronous model.

In order to compare the round complexity of an asynchronous execution of the modified protocol with a synchronous execution of invasion protocol, let us fix a sequence of random bits used by edges in a synchronous execution of invasion protocol, i.e., bits for performing populating operation. Let G be a given n-vertex landscape graph. Consider a vertex v and the path from the source of the invasion to vertex v through which populating v is performed in the considered execution of invasion with the fixed bits (i.e., to each vertex on this path, the first populated vertex comes through its predecessor on this path). Now consider an asynchronous execution of the modified protocol for the same set of random bits. Note that we fix only bits used for populating operation, while bits used to generate an asynchronous order of active edges are still random.

The probability that the number of rounds needed for populating vertex v through the same path in asynchronous execution of the modified protocol is asymptotically different than the number of rounds used in the corresponding synchronous execution (i.e., for the same fixed sequence of random bits for invasion) plus/minus $\log n$, is polynomially small. Here the probability distribution is over random bits used for selecting an asynchronous order of edges. This result holds by applying Chernoff bound and the fact that the probability of not skipping a round by a vertex on the path is at least $1 - (1 - 1/n)^n = \Omega(1)$. Taking a union bound of the above events, over all sequences of bits for invasion operation and over all edges m, we obtain the result claimed in the first part of the theorem.

In order to prove the second part, we observe that the modified protocol in asynchronous environment completes invading targets on landscape graph G is $O(\gamma(N))$ rounds, in expectation. Indeed, we enhanced the proof of Theorem 4.1 in a way to capture the skipped steps made by the modified protocol: an invasion step is skipped with a constant probability, as the expected number of repetitions of vertices within a round is a constant. This probability introduces an additional constant in front of all the formulas in the original proof of Theorem 4.1, which however does not change the asymptotic result of $O(\gamma(N))$ rounds, in expectation. Next we apply the relation between the round complexity of a synchronous execution of invasion and an asynchronous execution of the modified protocol, prove in the previous paragraph. This immediately implies the expected $O(\gamma(N) + \log n)$ round complexity of a synchronous execution of invasion protocol. ∎

3 Algorithms

Here we present Algorithms 1 and 2, together with Table S1 describing their input parameters and output variables.

Table S1: Input parameters and output variables for Algorithms 1 and 2.

Input parameters for Algorithm 1 and 2
G: 2D array stores qualities of patches in a given real landscape. S: vector contains indices of initial populated patches. T: vector contains indices of unpopulated target patches. α: given number > 0.
Output variables for Algorithm 1
Number of rounds needed for invasion. Execution time of simulation.
Output variables for Algorithm 2
Maximum flow for the invasion network. Execution time of computing max-flow.

Algorithm 1 Modelling invasion process using *full* simulation method (G, S, T, α)

```
function COUNT ROUNDS(G, S, T, α)
    start ← record start time of simulation
    Create 2-dimensional array B having size equal to array G ← 0
    for each index of populated patch in vector S do
        B(index) ← 1
    Rounds ← 0
    while all target patch in T is unpopulated do
        Rounds ← Rounds+1
        for i ← 0 to no. of rows in G do
            for j ← 0 to no. of columns in G do
                if patch B(i, j) is populated then
                    for z ← 0 to no. of rows in G do
                        for l ← 0 to no. of columns in G do
                            if patch B(z, l) is unpopulated and q(B) ≠ 0 then
                                p ← Transition probability between B(i, j) and B(z, l)
                                w ← Generate a random number between 0 and 1
                                if w < p then
                                    Populate patch B(z, l)
    end ← Record end time of simulation
    ExecutionTime ← end - start
    return Rounds, ExecutionTime
```

Algorithm 2 Modelling invasion process using network flow method (G, S, T, α)

function COMPUTE MAX-FLOW(G, S, T, α)
  start ← record start time of computing max-flow
  rows ← number of rows in G
  columns ← number of columns in G
  patches ← rows · columns
  $s \leftarrow 0$
  $t \leftarrow$ patches+1
  Create array $C[patches + 2, patches + 2] \leftarrow 0$
  for each non-zero patch in S **do**
    $C(s, patch) \leftarrow \lambda$
  for each non-zero patch in T **do**
    $C(patch, t) \leftarrow \lambda$
  for $i \leftarrow 1$ to patches **do**
    for $j \leftarrow 1$ to patches **do**
      $C(i, j) \leftarrow$ *Transition probability* between patches i and j
  max-flow← compute maximum flow (s, t, C)
  end ← Record end time of computing max-flow
  ExecutionTime ← end - start
  return Max-flow, ExecutionTime

4 Methods and additional simulation results

The average number of rounds over 2ST independent iterations (estimated invasion time) using *full* simulation and the proposed prediction methods for each prefix and in each landscape of size 5×300 and of low, medium, and high quality is presented in Figures S3, S4, and S5, respectively. In all types of quality, the invasion time increases with the increase in the prefix size. The increase in the dispersal coefficient α from 0.25 to 2 shows an increase in the invasion time in all qualities. The invasion time in landscape of low quality is longer than in landscapes of medium and high qualities. Therefore, the quality of the landscapes is an important factor on the invasion time.

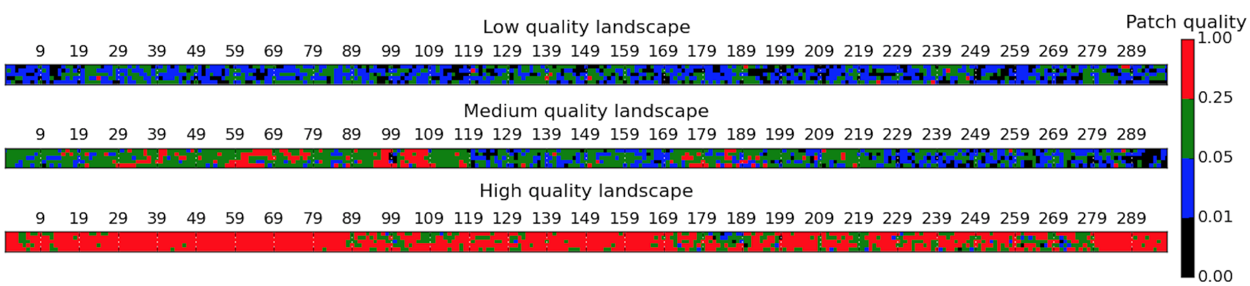

Figure S2: The studied landscapes. Three rectangular landscapes of size 5×299 and of low, medium and high quality extracted from LCM2007 UK maps (aggregate classes). In each landscape, the colour corresponds to the quality of each patch; black, blue, green, and red corresponds to zero, low (0.01-0.05), medium (0.05-0.25), and high quality (0.25-1), respectively.

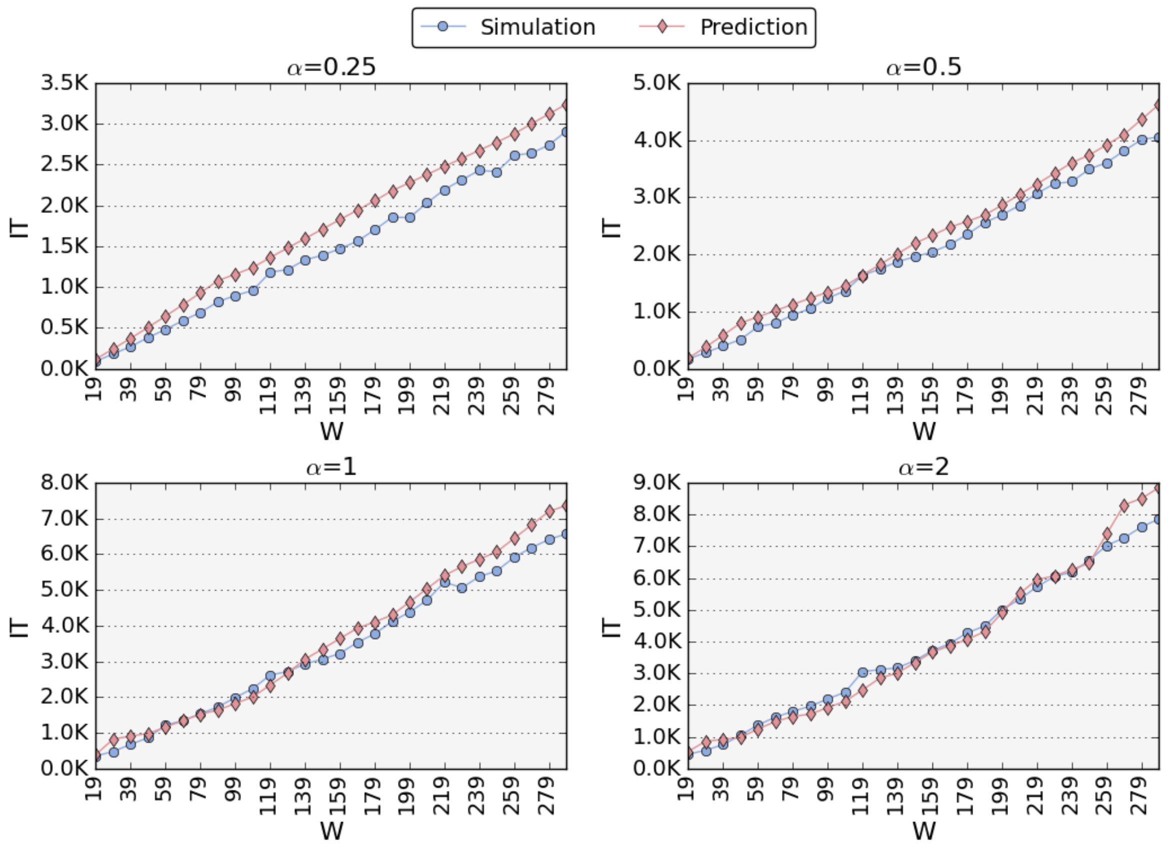

Figure S3: The invasion time IT using *full* simulation method and the proposed prediction method for each prefix in the landscape of size 5×299 and of **low** quality. This is done for different values of the dispersal coefficient α: 0.25, 0.5, 1 and 2.

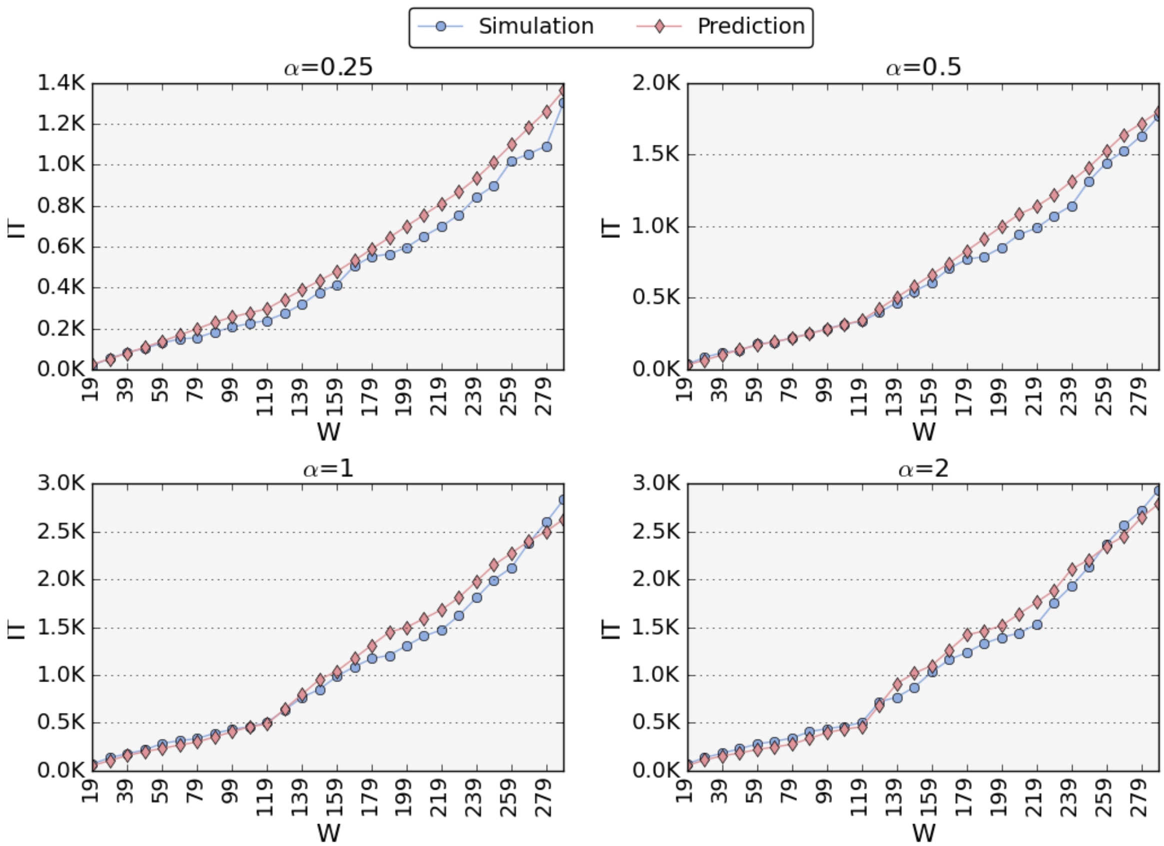

Figure S4: The invasion time IT using *full* simulation method and the proposed prediction method for each prefix of the **medium** quality landscape of size 5×299, for $\alpha = 0.25, 0.5, 1, 2$.

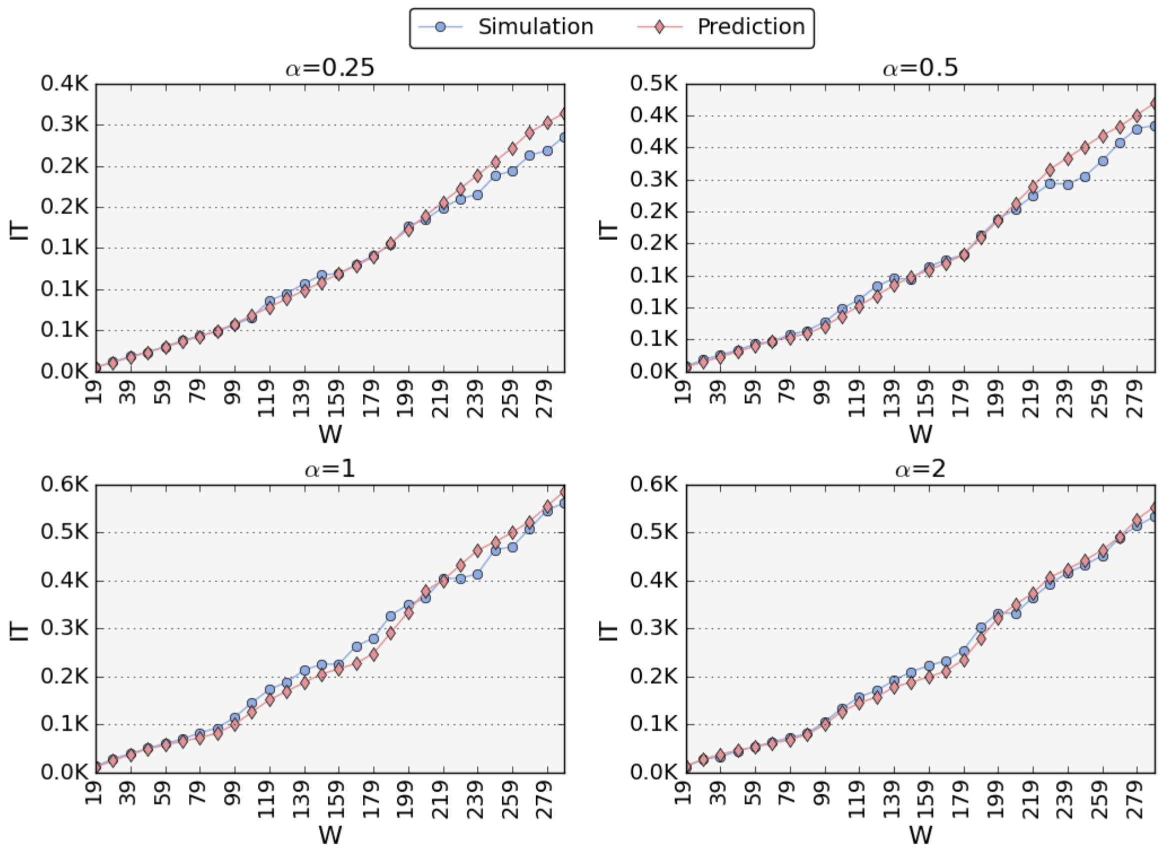

Figure S5: The invasion time IT using *full* simulation method and the proposed prediction method for each prefix of the **high** quality landscape of size 5×299, for $\alpha = 0.25, 0.5, 1, 2$.

5 Verification and improvement in time and memory

We present the mixed quality landscape (Figure S6) used for verification of the prediction formula (Equation 2 with Table 1 in the main paper) and measuring the saved time and memory. Figure S7 gives the average number of rounds over 2ST independent iterations (estimated invasion time) using the *full* simulation and the proposed prediction method for each prefix of the 10×299 landscape of mixed quality. Figure S8 shows how much memory is saved when using the proposed prediction method (Equation 2 in the main paper).

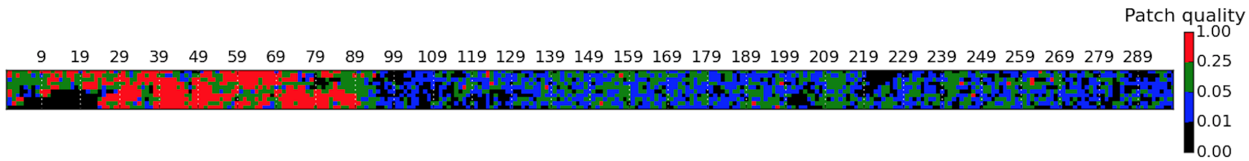

Figure S6: Mixed quality landscape of size 10×299 used to verify the new proposed prediction method. The colour corresponds to the quality of each patch; black, blue, green, and red corresponds to zero, low (0.01-0.05), medium (0.05-0.25), and high quality (0.25-1), respectively.

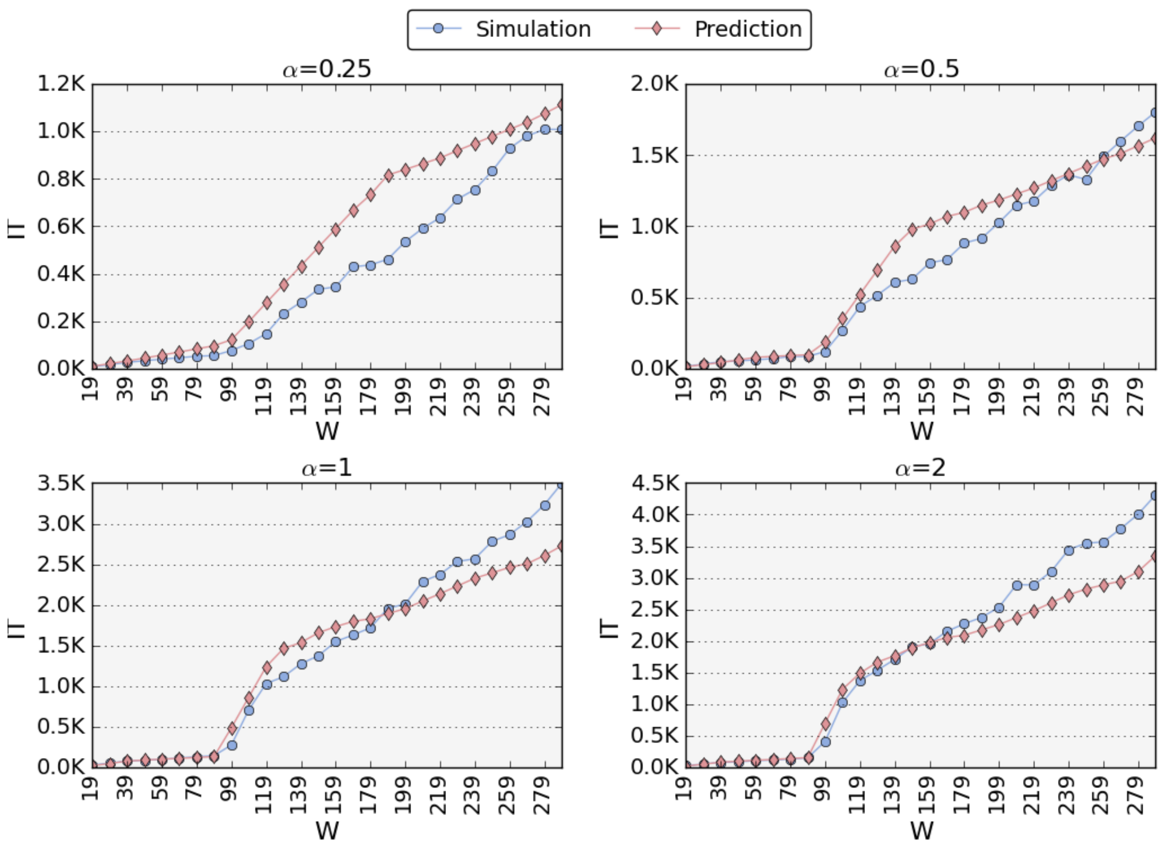

Figure S7: The invasion time IT using *full* simulation method and the proposed prediction method for each prefix in 10×299 landscape of **mixed** qualities. This is done for different values of α: 0.25, 0.5, 1 and 2.

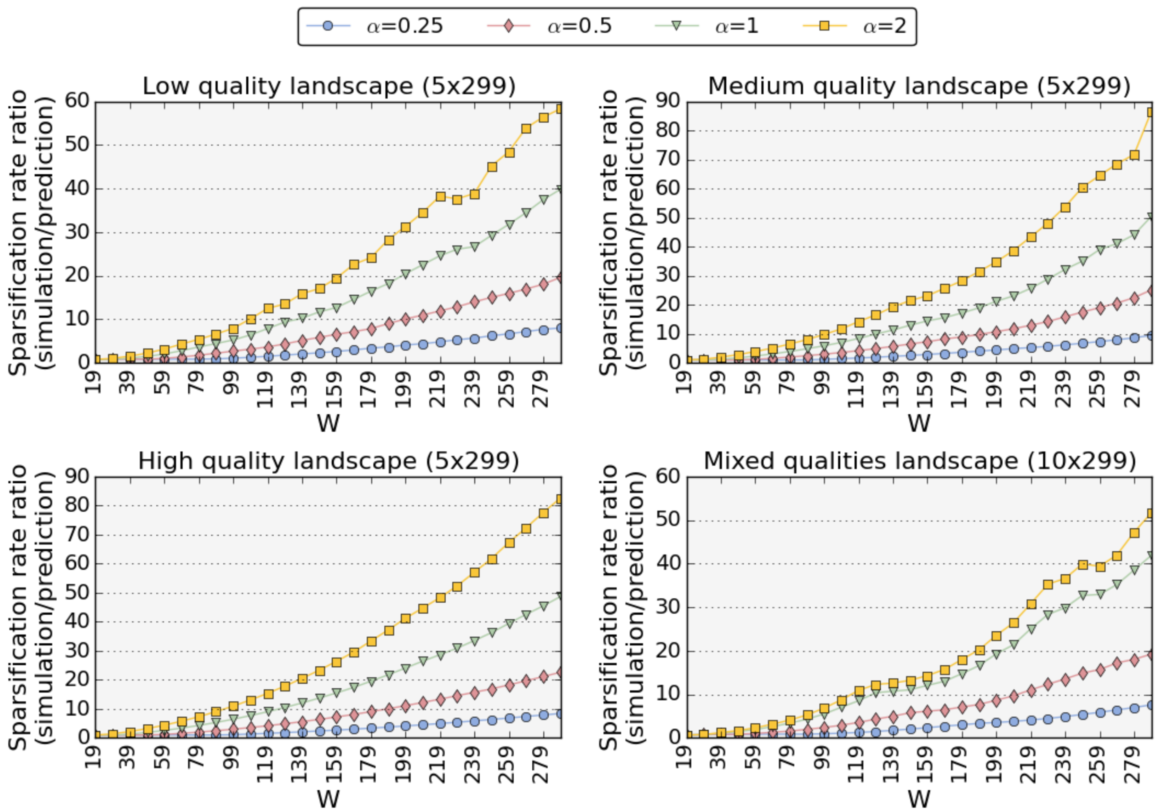

Figure S8: The ratio of the sparsification rate (simulation over prediction) versus the landscape width. That done for different values of α: 0.25, 0.5, 1 and 2, and for each landscape quality (low, medium, high, and mixed).